REMARKS/ARGUMENTS

This Amendment is in response to the Office Action mailed March 9, 2008. Claims 1, 5-16, 18, 19, and 24-36 are pending in the present application. Claims 1, 5, 6, 10, 12, 13, 15, and 29 have been amended. Claims 2-4, 17, and 20-23 have been canceled. New claims 34-36 have been added. No new matter has been added. Reconsideration of the rejected claims is respectfully requested.

I. Examiner Interview

A telephone interview was conducted with Examiner Alhija on April 2, 2008, at 12:30 PM PDT. The undersigned represented the Applicants in the interview. In the interview, differences between the claimed invention and the cited art were discussed. Although no agreement was reached, the Examiner indicated that amending the claims to further clarify differences between the claimed embodiments and the cited art would likely overcome the rejections and trigger a new search. Applicants appreciate the Examiner's helpful suggestions, and have amended the claims accordingly. Applicants believe that the claims as amended are allowable over the cited art.

II. Rejection under 35 U.S.C. § 102(b)

Claims 1, 5-12, 14-16, 18-19, 24-26, and 28-33 are rejected under 35 U.S.C. § 102(b) as being anticipated by *Lin* (US 6,389,379). Applicants respectfully submit that *Lin* does not disclose each element of these claims.

Claim 1

For example, Applicants' claim 1, as amended, recites a method in a host machine for validating a design for a system which comprises a software element and first and second hardware components, the software element being for execution on the second hardware component and the first and second hardware components being operable to interact with one another, the method comprising the steps of:

> **simulating operation of the first hardware component in a first software simulation system;**

> **simulating the software element and the second hardware component in a second software simulation system;**
>
> **receiving a variable synchronization parameter;**
>
> **running the second software simulation system asynchronously with, and ahead of, the first software simulation system, wherein the second software simulation system advances at most by a number of processor clock cycles set in the variable synchronization parameter**, the variable synchronization parameter limiting a maximum number of processor clock periods of the second simulation per period of a reference clock of the host machine;
>
> controlling the first software simulation system using the second software simulation system that is running ahead of the first software simulation system , a socket allowing for communication between the second software simulation system and the first software simulation system; and
>
> analyzing a result from the first and second software simulation systems and validating the design for the system,
>
> wherein the first software simulation system and the second software simulation system are implemented in separate processing threads within the host machine providing more rapid simulation of software instructions in the second software simulation system than the simulation of instructions in the first software simulation system

(*emphasis added*).

i. *<u>Lin</u> does not disclose running the second software simulation system asynchronously with, and ahead of, the first software simulation system.*

The pending Office Action on page 3 states that the "SEmulation system allows both asynchronous and synchronous data inputs depending upon which is enabled." The Office Action also asserts Figure 17 as supporting asynchronous operation. However, Figure 17 and *Lin*, column 56, lines 37-47 disclose a hardware register (e.g. a D-type flip-flop). This register is a basic building block of the **hardware** model, but there is no disclosure in *Lin* for a **second software simulation system** running **asynchronously** with, and ahead of the first software simulation system, as required by claim 1. While the single register in Figure 17 may receive asynchronous data, the accompanying passage in *Lin* fails to disclose, and the Office Action does not explain, how receiving asynchronous data in a single hardware register equates to, "running the second software simulation system asynchronously with, and ahead of, the first software simulation system."

Also, the pending Office Action on page 2 asserts that Figure 49 of *Lin* "reads on one part of the simulation running ahead of or behind another." Figure 49 discloses a flow chart of the operation of a simulation server that allows multiple users to share the simulation system on a

time-shared basis. (*See Lin*, col. 88, lines 15-20, and col. 91, lines 59-61.) The simulation system swaps between **different** jobs (i.e. designs or users) based on priority and availability of the simulation system. (*See Lin*, col. 86, lines 18-21.) The simulation system in *Lin* also provides for two simultaneous users because *Lin* discloses a system that simulates a design using **one mode of simulation at a time** (e.g. either software simulation or hardware acceleration). *Lin* discloses that a first user can use the software simulator for a first design, and a second user can use the hardware simulator for a second design.

In addition, assuming that there is no second user or that the first user has priority over the second user, the first user is able to switch between software and hardware modeling of a design because *Lin* discloses simulating a design using one mode of simulation at a time. (*See Lin*, col. 17, lines 45-50 and col. 18, lines 20-31.) For example, *Lin* discloses, "the system can simulate the circuit in software for a time period, accelerate the simulation through the hardware model, and return back to software simulation mode." (*Lin*, col. 18, lines 28-31.) The design simulated in *Lin* uses one simulation system (e.g. hardware or software) at a time, so *Lin* does not disclose simulation of a design using both the hardware and software simulations in parallel.

Even if *Lin* is seen as disclosing two software simulations running with each other, neither simulation in *Lin* can run ahead or behind the other because there is no reference between the two simulations that would allow for one simulation to be perceived to be ahead or behind the other. In other words, even if there is one design being simulated in the software simulator and a second design being simulated on the hardware simulator at the same time, the designs cannot be "ahead or behind" because there is **no interaction** or reference between the two simulations that would allow the perception that one simulation is running ahead (or behind) the other. Each simulation in *Lin* is independent, and *Lin* does not disclose any connection between the two simulations that would provide such a reference. (*See Lin*, col. 92, lines 42-65.) In contrast, claim 1 recites a design comprising first and second hardware components **operable to interact with one another**, wherein the first and second hardware components are simulated in the first and second software simulation systems, respectively. Thus, *Lin* does not disclose

"running the second software simulation system asynchronously with, and ahead of the first software simulation."

ii. *<u>Lin</u> does not disclose receiving a variable synchronization parameter.*

Claim 1 recites, "receiving a variable synchronization parameter" and "running the second software simulation ... wherein the second software simulation system advances at most by the number of processor clock cycles set in the variable synchronization parameter." *Lin* does not disclose these limitations. As discussed above, *Lin* is directed to a system that simulates a circuit using one mode of simulation at a time (e.g. either software simulation or hardware acceleration). Thus, there is no need in *Lin* for a parameter to limit advancement of the hardware simulation, in relation to the software simulation, since the simulations do not interact with each other. Therefore, *Lin* does not disclose receiving a variable synchronization parameter.

iii. *<u>Lin</u> does not disclose first and second software simulation systems.*

Lin does not disclose a **first software** simulation system and a **second software** simulation system as recited in claim 1. Instead, as seen on Figure 67, *Lin* discloses a simulation system that uses **a software simulation system** and **a hardware simulation system** rather than two software simulation systems. In particular, Figure 67 discloses, "the RCC computing system 2081 includes the entire model of the user's design in software and the RCC hardware array 2084 includes a hardware model of the user's design." (*See Lin*, col. 123, lines 38-41.) *Lin* models the hardware portion of the design using hardware components, so a second software system is unnecessary.

As explained in *Lin*, there are generally two major types of coverification tools. One type of coverification tool uses the actual hardware in conjunction with a software simulator to test a system, while the other type uses two software simulators – one for software emulation and one for hardware emulation. (*See Lin*, col. 4, lines 56-57.) *Lin* discloses the former type, and uses FPGA chips for the hardware modeling. *Lin* describes the hardware as,

> The RCC hardware array system 2084 includes a PCI interface 2085, a set of RCC hardware array boards 2086, and various buses for interface purposes. The set of RCC hardware array boards 2086 includes at least a portion of the user's design modeled in hardware (i.e.,

> hardware model 2087) and memory 2088 for the test bench data. In one embodiment, various portions of this hardware model are distributed among a plurality of reconfigurable logic elements (e.g., FPGA chips) during configuration time.

(*Lin*, col. 123, lines 49-57.) *Lin* operates by modeling the entire system in software as well as hardware. (*See Lin*, col. 123, lines 38-41.) However, simulating some hardware components in software is slow, so *Lin* provides hardware modeling using FPGAs to accelerate portions of the simulation. (*See Lin*, col. 120, line 64 to col. 121, line 5.) Thus, *Lin* does not provide a need for a second software simulation system.

For at least these reasons, *Lin* does not anticipate claim 1, and withdrawal of the rejection is respectfully requested.

Claim 5

Dependent claim 5 recites the method of claim 1 further comprising:

> performing operations in the first software simulation system to set up an inter-process communications protocol connection therein;
> connecting the second software simulation system to the inter-process communications protocol connection in the first software simulation system;
> connecting a **software debugger** to the second software simulation system; and
> controlling the first software simulation system from the **software debugger** via the second software simulation system using the inter-process communications protocol

(*emphasis added*).

These features are not disclosed in *Lin*. For example, *Lin* does not disclose "a software debugger." A word search found no reference to any software debugger in *Lin*. However, the Office Action refers Applicants to *Lin* column 1, lines 33-49 as well as the abstract as disclosing this feature, but this feature is not disclosed in the cited passages, which are included below:

> The value of software simulating a circuit design followed by hardware emulation is recognized in various industries that use and benefit from EDA technology. Nevertheless, current software simulation and hardware emulation/acceleration are cumbersome for the user because of the separate and independent nature of these processes. For example, the user may want to simulate or **debug** the circuit design using software simulation for part of the time, use those results and accelerate the simulation process using hardware models during other times, inspect various register and combinational logic values inside the circuit at select times, and return to software simulation at a later time, all in one **debug**/test session. Furthermore, as internal register and combinational logic values change as the simulation time advances, the user should be able to monitor these changes even if the changes are occurring in the hardware model during the hardware acceleration/emulation process.

(*Lin*, col. 1, lines 33-49.)

> The coverification system includes a reconfigurable computing system (hereinafter "RCC computing system") and a reconfigurable computing hardware array (hereinafter "RCC hardware array"). In some embodiments, the target system and the external I/O devices are not necessary since they can be modeled in software. In other embodiments, the target system and the external I/O devices are actually coupled to the coverification system to obtain speed and use actual data, rather than simulated test bench data. The RCC computing system contains a CPU and memory for processing data for modeling the entire user design in software. The RCC computing system also contains clock logic (for clock edge detection and software clock generation), test bench processes for testing the user design, and device models for any I/O device that the user decides to model in software instead of using an actual physical I/O device. The user may decide to use actual I/O devices as well as modeled I/O devices in one **debug** session. The software clock is used as the external clock source for the target system and the external I/O devices to synchronize all data that is delivered between the coverification system and the external interface. The coverification system contains a control logic that provides traffic control between: (1) the RCC computing system and the RCC hardware array, and (2) the external interface (which are coupled to the target system and the external I/O devices) and the RCC hardware array. Because the RCC computing system has the model of the entire design in software, including that portion of the user design modeled in the RCC hardware array, the RCC computing system must also have access to all data that passes between the external interface and the RCC hardware array. The control logic ensures that the RCC computing system has access to these data. Pointers are used to latch data from the RCC computing system and the external interface to the internal nodes of the hardware model in the RCC hardware array. Pointers are also used to deliver data from the internal nodes of the hardware model to the RCC computing system and the external interface. Even if the data from the internal nodes of the hardware model is intended for the external interface, the RCC computing system must also be able to access this data as well.

(*Lin*, Abstract.)

While both of these passages disclose the need to debug a design during a "debug session," which is the purpose of using a coverification simulator, neither passage discloses a "software debugger." Even if there was a suggestion for a "software debugger," *Lin* does not disclose "connecting a software debugger to the second software simulation system; and controlling the first software simulation system from the software debugger via the second software simulation system using the interprocess communications protocol."

Therefore, *Lin* does not anticipate claim 5, and withdrawal of the rejection is respectfully requested.

Claims 6-11, 14-16, 18, 19, 24-26, and 28-33

Claims 6-11, 14-16, 18, 19, 24-26, and 28-33 recite limitations that similarly are not disclosed by *Lin* for reasons including those discussed above. As such, these claims are also not anticipated by *Lin*. Therefore, withdrawal of the rejection of these claims is respectfully requested.

III. Rejection under 35 U.S.C. § 103(a)

Claims 13 and 27 are rejected under 35 U.S.C. §103(a) as being obvious over *Lin* in view of *Kim* ("An integrated Hardware-Software Cosimulation Environment with Automated Interface Generation").

Combining *Kim* with *Lin* would not make up for the deficiencies in *Lin* with respect to these claims. *Kim* teaches a co-simulation environment (Abstract; Introduction) and is cited on page 15 of the Office Action as teaching the use of a C model to implement a second simulation. Even if the teachings of *Kim* were combined with those of *Lin*, the resultant combination would fail to teach or suggest "receiving a variable synchronization parameter." Nor would the combination teach or suggest, "running the second software simulation system asynchronously with, and ahead of, the first software simulation system, wherein the second software simulation system advances at most by a number of processor clock cycles set in the variable synchronization parameter, the variable synchronization parameter limiting a maximum number of processor clock periods of the second simulation per period of a reference clock of the host machine."

A combination of *Kim* with *Lin* thus would not arrive at such limitations, even assuming for sake of argument that there were motivation to combine the references. Therefore, *Kim* and *Lin* cannot render obvious Applicants' claims 1 or 15, or dependent claims 13 and 27, individually or in combination. Applicants therefore respectfully request that the §103 rejections with respect to claims 13 and 27 be withdrawn.

III. Amendments to the Claims

Unless otherwise specified or addressed in the remarks section, amendments to the claims are made for purposes of clarity, and are not intended to alter the scope of the claims or limit any equivalents thereof. The amendments are supported by the specification and do not add new matter.

CONCLUSION

In view of the foregoing, Applicants believe all claims now pending in this Application are in condition for allowance and an action to that end is respectfully requested.

If the Examiner believes a telephone conference would expedite prosecution of this application, please telephone the undersigned at 925-472-5000.

Respectfully submitted,

Charles F. Koch
Reg. No. 58,669

TOWNSEND and TOWNSEND and CREW LLP
Two Embarcadero Center, Eighth Floor
San Francisco, California 94111-3834
Tel: 925-472-5000
Fax: 415-576-0300
C1K:km
61315357 v1